CS 1337 Computer Science I
Fall 2018 Syllabus UTD

INSTRUCTOR INFORMATION

Name	Mr. Jason Smith
E-mail Address	jws130830@utdallas.edu
Facebook	http://www.facebook.com/MrSmithUTD
Office	ECSS 3.232
Office Phone	972-883-4835

Office Hours:	M/W	1:30 – 2:15 PM
	Tu/Th	11:30 AM – 12:30 PM
		2:30 – 3:00 PM

No appointment necessary!
Stop by to ask questions, chat or play a game of chess

Questions: Instead of emailing me your questions, please post them here: http://piazza.com/utdallas/fall2018/cs1337/home

COURSE INFORMATION

Course Number	CE/CS/TE 1337.001 and .002		
Credit Hours	3		
Meeting Time	001	M/W	2:30 – 3:45 PM
	002	Tu/Th	1:00 – 2:15 PM
Room	001	ECSW 4.325	
	002	ECSS 2.305	

DO YOU NEED ASSISTANCE?

Piazza: The easiest way to get help is by posting your questions on Piazza, a new platform that will help you get answers to your questions quickly. You will be able to post your questions (anonymously if you wish) about anything related to the class (except grades) and get a response either from me or a classmate. Since there are multiple people that can answer questions, you should get a quicker response allowing you to complete the tasks you are working on.

Help Desk: For help with issues regarding your computer, UTD maintains a walk-in help desk. Visit their Web site for details: http://www.utdallas.edu/ir/helpdesk/

Tutoring: For programming assistance in CS1337, please visit me, the TA, or the Mentoring Center. The schedule for the Mentoring Center will be released within the first week of classes. Once the Mentoring Center schedule for this semester has been released, an announcement will be posted on eLearning. **If you need help, please make the effort to reach out. We can't help you if we don't know that you need help.**

Resources:

- C++ language tutorial http://www.cplusplus.com/files/tutorial.pdf
- C++ reference: http://www.cppreference.com
- C++ tutorial http://www.learncpp.com/

WHAT DO I NEED FOR CLASS?

- **Notebook –** You are going to take a lot of notes and have quizzes. Bring paper to write on.

- **Writing Tool** – pen, pencil, crayon, etc. It's hard to take notes without one. You are free to use your own blood, but that gets messy.

- **Textbook**:
  - **Option 1:** Revel System
    - Revel for Gaddis C++ - Access Code (Digital Book)
      Gaddis, Tony; Pearson Publishing
      ISBN 0-13-449837-9

    - The access code is for a digital version of the book using the REVEL platform. The REVEL platform provides interactive elements to the digital textbook such as animations, videos and coding samples that readers can modify and execute.
      - In past semesters, students have stated that the interactive textbook helped them learn the material better

    - Students using this option do not need to buy the physical version of the book
    - Extra credit opportunities **may** become available through the Revel system

  - **Option 2**: Physical book
    - Starting Out with C++, From Control Structures through Objects
      (9th edition – grapefruit slice) ; Gaddis, Tony; Pearson Publishing
      ISBN 0-13-449837-9

    - As you read the text, watch the corresponding VideoNotes. The VideoNotes are available at http://www.pearsonhighered.com/gaddis/.
      NOTE: VideoNotes are only available with an access code. If your book does not have an access code, you can buy one online at the above address. **The access code is not required for class**, but some of you may find the material accessible with this code to be a good resource.

    - Students may use the 8th edition of the book. In doing so, students accept the responsibility of verifying page numbers for assignments as well as learning the topics not present in the 8th edition. I will not copy any information from the 8th edition for students, although you are free to take pictures of pages from my book during office hours.

- **C++ Compiler (Required)**
  - All projects you submit will be compiled with gcc/MinGW-w64 v7.3.0. You may use any IDE that can utilize gcc 7.3.0.

    MinGW can be downloaded from the following link:
    https://sourceforge.net/projects/mingw-w64/files/Toolchains%20targetting%20Win64/Personal%20Builds/mingw-builds/7.3.0/threads-win32/sjlj/x86_64-7.3.0-release-win32-sjlj-rt_v5-rev0.7z/download

- In class, the IDE I will be using is Code::Blocks 16.01. This is a free download for Windows. However, you may use any IDE you choose as long as it supports the proper version of gcc.
  http://sourceforge.net/projects/codeblocks/files/Binaries/17.12/Windows/codeblocks-17.12-setup.exe

- If you choose to use Code::Blocks, there is a document in the Course Resources section in eLearning to link the compiler to the IDE.

- For Mac users, I recommend using XCode or creating a Windows partition to install MinGW and an IDE. Be advised that there is a Mac version of Code::Blocks, but it has been others have reported it doesn't work with newer versions of the OS X operating system.

  Please note that XCode's default compiler is not gcc. It is highly recommended that you test your code with gcc prior to submission or learn how to change the default compiler to gcc.

- **If a student uses a compiler other than gcc 7.1.0 for development, he/she is responsible for verifying prior to submission that the code compiles and executes properly with the stated compiler.** No compiler is perfect and each one has its own quirks. It is the student's responsibility to make sure that the program functions as expected with the compiler that will be used for grading (gcc 7.1.0).

- If you intend to use your own computers to write the class assignments, it is important that you get a compiler downloaded, installed, and running on your computer as soon as possible. If you don't have a computer, or if you're having problems getting a compiler installed, you should write your programs in the labs until the problems are resolved. In any case, please note that you are responsible for getting the programming assignments written and turned in on time. Since there are many computers available on campus, problems with your local machines will not be accepted as an excuse for not doing the assignments or late submissions.

REQUIRED COURSE INFORMATION SECTION

Course Prerequisite: Prerequisite: CS 1336 with a grade of C or better or equivalent (placement test or AP credit).

Description of Course Content: Review of control structures and data types with emphasis on structured data types. Applies the object-oriented programming paradigm, focusing on the definition and use of classes along with the fundamentals of object-oriented design. Includes basic analysis of algorithms, searching and sorting techniques, and an introduction to software engineering.

Student Learning Outcomes: After successful completion of this course, the student should have an:

- Ability to use single and multi-dimension arrays.
- Ability to implement linear and binary searches.
- Ability to implement simple sorting algorithms.
- Ability to implement structured data types.
- Ability to define and implement a class.
- Ability to use fundamentals of object-oriented design.

Departmental Attendance Policy: The Computer Science Department has implemented the following attendance policy beginning Fall 2016. **If a student misses three consecutive classes, the student will receive a letter grade reduction to his or her final grade.** This deduction is cumulative, so if a student misses three consecutive classes twice, the final grade will be reduced by two letter grades. **If a student misses four consecutive classes, the student will automatically receive an F for his or her final grade.**

WHAT I EXPECT OF EACH STUDENT

- **Ask for help at any time.** If you do not understand something or are having trouble implementing a concept, reach out. During lecture, before/after class, during office hours, at 2 AM after a night clubbing the day before something is due, etc. I really mean any time. The sooner you ask that question, the sooner you will get an answer. That answer will allow you to move forward. I want you to succeed; don't be afraid to ask a question.

- **Take responsibility for your education.** I will treat this course as similar to a professional setting as I can. I am not the type of teacher that lectures with slides and expects students to memorize. I will teach by creating program examples in class. I will give challenging assignments to push you toward learning the intricacies of C++.

  Part of being a professional is learning how to teach yourself. I am going to guide you through the topics of the semester, but a significant portion of what you take with you to the next class will be things that you learned on your own.

- **Practice time management skills.** All assignments (homework and projects) are designed to be worked on over a period of days or weeks. I expect that you will work on the assignment a little at a time rather than waiting until a day or two before it is due. **Those that procrastinate will find this class to be much harder than it should be and will face the risk of below average grades.**

- **Attend every class.** You are paying for an education. Don't waste your money by skipping class.

- **Make mistakes!** This is how you learn how to do something. Don't be discouraged when something goes wrong. Programming takes lots of practice and mistakes will always happen. Study the mistakes you made so that you can learn the correct way to do it.

- **Read the chapter before the corresponding lecture** (see class schedule below)**.** I use class time to write programs that help illustrate the topics mentioned in each chapter. If a student doesn't have minimal knowledge of the concepts that will be covered for that chapter (which are gained by reading the chapter), it will be harder to get a deeper connection to what we are accomplishing in class.

- **Arrive to class on time and remain in class until dismissed.** Arriving late and leaving early cause disruptions to the other students in the class and to me. Should you need to leave early for a valid reason, please notify me in advance and sit near the door to limit the disruption. Don't make me call you out.

- **Don't sleep in class.** Let's be honest; programming in C++ is not the most exciting topic. Combine that with fatigue from late night gaming and/or study sessions and it is super easy to doze off. Fight it off. Bring in a caffeinated beverage of your choice, such as Starbucks coffee or a Monster energy drink. Carry an emergency bottle of 5 Hour Energy in your backpack if need be. If students could learn C++ by sleeping, there would be no reason to get out of bed to attend class.

- **Don't pack up your things until class is over.** Most of the time we will go until the very last minute before ending class. Sometimes we might go over by a minute or two if I need to finish a discussion. If students start packing up before we are finished, it makes a lot of noise. That noise might prevent someone from hearing very crucial information such as what the next homework assignment is and when it is due. It also makes me think you are in a hurry to leave the awesome fun party we are having and hurts my feelings.

- **No computers in class.** I know, this sounds like crazy talk to say no computers in a computer science classroom, but hear me out. I have seen grades improve by about 10% in classes where I do not allow computers. Without a computer in front of them, students are more engaged during class time.

  Tablets or laptops that can be converted into tablet mode are allowed for students to take notes.

  Some students want to code during class, however, this becomes a detriment to the student as well as those seated around that person. While students are copying the code that I am writing in class, they are not concentrating on the logic or details behind the code. It is the logic and ideas behind the statements that are more important. All code that is written in class will be posted in eLearning after it is completed so that you can play with it.

WHAT EACH STUDENT SHOULD EXPECT

- **A problem solving class.** This class is not a programming class. Computer science is all about problem solving. The content of this class is to teach you how to solve problems using a computer. In order to solve those problems, you will need to learn a foreign language (C++) and write solutions that the computer can interpret.

- **An open environment dedicated to learning.** I want students to feel free to voice their opinions. Oftentimes as we code in class, I will ask students what they would do in a certain situation. I want each student to feel as if he/she can speak freely and also be open for other students to discuss that idea, even if that means that some students will disagree.

- **Class commitment of 10-12 hours a week on average outside of class.** Students should be prepared to tackle multiple course-related activities each week (e.g. reading the textbook, studying for quizzes/exams, practicing programming, etc.). Procrastinating on an assignment will largely increase the number of hours spent each week.

- **Exams focused on application.** I do not create run-of-the-mill multiple choice exams that ask students to regurgitate things from memory. The exams are completely different than anything you have had in any other class (unless you've had me for a previous class). I expect you to apply the knowledge you have learned to the situations on the test. Questions on the test are designed to make sure that you understand what you are doing rather than repeating an example from your notes or the textbook.

- **A simulated professional experience.** The projects in this class require you to exercise strategies found in "the real world". Your logic for a project may force you to learn new techniques that haven't yet been discussed in class. You will have to perform code maintenance and improve the efficiency of previously written code. These things offer a small taste of how life might be once you graduate and are given large sums of money by a company seeking your skills.

- **A deep understanding of C++.** My goal is for you to know all of the topics of CS 1337 as well (if not better) than me, and I'm going to push you toward that goal. You should have

peace of mind moving on in your program because you will be fully prepared to tackle what the next course in the sequence will throw at you.

THE INFORMATION YOU REALLY CARE ABOUT

Grading Scale:

98-100 A+	88-89 B+	78-79 C+	68-69 D+	Below 60 F
92-97 A	82-87 B	72-77 C	62-67 D	
90-91 A-	80-81 B-	70-71 C-	60-61 D	

Grade Components:

Projects (4)	50%
Exams (3)	30%
Preview Homework	5%
Review Homework	5%
In-class Assignments	10%

General Grade Information: All grades will be available in eLearning. The Weighted Total column will give you the most accurate information concerning your grade. The weighted total is an approximation of your grade in the class based on the grades currently in eLearning.

I do not curve grades. Assignments are combined into categories so that a low grade for one item will not destroy your grade. There are also opportunities provided to help students who may have done poorly on projects and exams.

Grade Disputes: All grade disputes must be reported within 1 week and resolved within 2 weeks of the grade in question being posted in eLearning. Uncontested grades will become final after 1 week and cannot be disputed later. Announcements are made after each grade is posted so please check your grades promptly and reach out to the proper person.

I am responsible for grading your exams. If you have questions regarding your exam, please contact me. Please note that due to FERPA, I cannot discuss grades via e-mail.

Everything else will be graded by a TA. Please address any grading concerns you have regarding these grades with the TA. **When you email the TA with questions about your grade, copy me on the email so that I am aware of the situation and can make sure it is resolved.**

Project Compilation Errors: There are times that the TA is unable to compile a project. When this happens, the TA will write a comment for the project submission in eLearning stating the project didn't compile. If you believe this to be an error, please meet with the TA (either during the TA's office hours or through a scheduled appointment) to demonstrate that the code does work on your PC. You must either download the code submitted to eLearning or verify that the code has not been modified since the due date. In addition, you must demonstrte to the TA that you are compiling with gcc 7.1.0.

Project Regrades: Frequently, students will have small logical errors in the code that cause points to be deducted from the project grade. These errors typically require a small change in the code (1 or 2 lines) to fix the problem. Students may fix these issues and have the project regraded.

Students are limited to having 1 project regraded during the semester. All project regrades must be conducted in my office. After each project's due date, a list of available times will be

posted for you to schedule an appointment for the regrade. Any student initiating a project regrade must submit a log of the changes in eLearning prior to meeting with me. You may fix any errors that are currently present in the code, but **you may not add new logic into the program**. For example, if a project requires you to sort data and you did not originally have a sorting algorithm implemented, you cannot add one for the project regrade.

The regrade only applies to the results of the test cases. The results of the test cases comprise 60 points of each project grade. Students can earn half of the points missed on the test cases (a maximum of 30 points) by using the regrade if the project works as expected.

Late Assignments: Homework and In-Class Assignments will not be accepted late. Homework assignments are to be submitted before the start of class on the due date. In-class assignments are due at the end of class on the day assigned.

Late Projects:
All projects will be due at 10 PM on the day listed in the project documentation. Projects will be accepted late at the penalty of 5% per hour late (rounded up) for up to 5 hours past the due date/time. This is only for the actual code you submit. Pseudocode is not accepted late.

Projects: Projects will be major programming assignments that supplement recently discussed topics and will be completed in two to three weeks. Projects are intended to take approximately 15-20 hours to complete overall; this includes the design, coding and testing process. Waiting until a couple of days before the due date to start the project is a bad idea. Not only does this introduce unnecessary stress into your life, it hardly ever ends well for the student. Most students score poorly on projects that are built in less than three days.

Projects are individual endeavors and students are not to work in groups on any project. Students are permitted (and I openly encourage students) to discuss the project. Feel free to share ideas on the logic, but **DO NOT SHARE ANY CODE.** When discussing logic, try to keep it general. If you give out every little piece of logic you have, there is a good chance the person you are helping will have very similar code as yours and may be flagged for being too similar. Be careful of posting your code online. Another student could use your code without your knowledge and could involve you in a code plagiarism referral.

Students should avoid using web sites like GitHub and Chegg for help on projects. Copying code from a web site is considered plagiarism and will be treated as such. If you find code on a web site, it is highly likely another student will find it as well which may cause both submissions to be flagged for similarity.

All projects will be submitted in eLearning and will be compared for originality. Any projects that are approximate or identical copies will be reported to the Office of Community Standards and Conduct, and I will accept their decision in regards to the grade if they believe that academic dishonesty has occurred.

Programming assignments will be graded on a 100 point basis. Not only will your project be graded on proper execution, but also things like efficiency, implementation and documentation. Keep in mind that you always want to write code that is easy to understand and is also easy to maintain. Fewer lines do not necessarily mean a better program. Please use comments liberally.

You are responsible for testing your project thoroughly before submission. I will not give you the test cases that will be used for grading before the project is due. As a computer scientist, you must be able to identify all possible input and make sure that your code produces proper output and does not crash.

Preview Homework: Preview homework assignments are questions based on the reading. These assignments will be in the form of quizzes in eLearning. Students will read the chapter and answer basic questions about the material to illustrate how well they understand the general concepts.

Review Homework: Review homework assignments are generally short coding assignments that can be done in 1-2 hours that measure how well you understand the material we have covered. These assignments are typically due 1 week from the date given.

In-class Assignments: In-class assignments are given periodically during the semester. Each in-class assignment will involve working in pairs. Each assignment will be introduced at the beginning of class and students will be expected to complete the assignment by the end of class. Students must be present in class to receive credit for the assignment.

Exams: Exams will cover chapters as listed below in the tentative course schedule. Exams will include a variety of question types including multiple choice, multiple answer and essay questions. Students are expected to be able to apply knowledge from all previous chapters in conjunction with the tested chapters. Exams are not created to make you feel smart; they are designed for you to demonstrate your understanding of the concepts. A high score on an exam exhibits a deep understanding of the topics.

An exam should not be missed except for the most extreme circumstances (such as hospitalization or death of an immediate family member). If you miss an exam, you must have documentation for the absence. A make-up exam may be given to students with valid documentation. The allowance of a make-up exam is at the sole discretion of the instructor.

ARE WE THERE YET?

All dates are subject to change at the discretion of the instructor

Date (001)	Date (002)	Topic	Reading Assignments
8/20	8/21	Introduction to CS 1337 Introduction to Debugging	Read Chapter 10
8/22	8/23	Intro to C Character and C-string Functions	
8/27	8/28	Characters, Strings and the String Class	Read Chapter 12 (omit 12.7, 12.8)
8/29	8/30	Advanced File I/O	
9/3		**LABOR DAY - NO CLASS (001)**	
9/5	9/4	Advanced File I/O	Read Chapter 19
9/10	9/6	Recursion	
9/12	9/11	Quicksort	
9/17	9/13	**In-Class Assignment #1**	Read Chapter 9
9/19	9/18	Pointers	**Project 1 Due**
9/24	9/20	**Exam 1 (Chapters 10, 12, 19)**	
9/26	9/25	Pointers	
10/1	9/27	**In-Class Assignment #2**	Read Chapter 11
10/3	10/2	Structured Data	
	10/4	**CONFERENCE DAY – NO CLASS (002)**	
10/8	10/9	Structured Data with Pointers	Read C. 17.1 & 17.2 Review Linked List PDF
10/10	10/11	Linked Lists	**Project 2 Due**
10/15	10/16	Enumerated Data Types Unions	
10/17	10/18	**In-Class Assignment #3**	Read Chapter 8

10/22	10/23	Searching and Sorting Arrays	
10/24	10/25	Searching and Sorting Arrays	
10/29	10/30	**Exam 2 (Linked Lists, C. 8, 9, 11)**	Read Chapter 13
10/31	11/1	Introduction to Classes	**Project 3 Due**
11/5	11/6	Introduction to Classes	
11/7	11/8	**In-Class Assignment #4**	Read Chapter 14
11/12	11/13	More About Classes	
11/14	11/15	More About Classes	Read Chapter 15
11/19 – 11/23		**Thanksgiving Break (No Classes)**	
11/26	11/27	**In-Class Assignment #5**	
11/28	11/29	Inheritance, Polymorphism & Virtual Functions	**Project 4 Due**
12/3	12/4	Inheritance, Polymorphism & Virtual Functions	
12/5	12/6	**In-Class Assignment #6**	
12/13	12/13	**Exam 3 (C. 13, 14, 15)**	

Important Dates:

August 20	Classes start
September 3	Labor Day
September 5	**Census Day – Last day to withdraw without a W**
September 18/19	Project 1 Due
September 20/24	Exam 1
October 10/11	Project 2 due
October 29/30	Exam 2
Oct. 31/ Nov. 1	Project 3 due
November 5	**Last day to withdraw**
November 28/29	Project 4 due
December 13	Exam 3

The above schedule is subject to change at the discretion of the Professor.

University Policies

For all other University policies, please visit http://go.utdallas.edu/syllabus-policies